Our Ref.: 493-37-3

U. S. PATENT APPLICATION

Inventor: Robert M. BEST

Invention: Copy Protection of Portable Game Software

Suggested class/subclass: 463-44 and 705-57

Graybeal Jackson Haley LLP
Attorneys at Law
155 108th Avenue NE, Suite 350
Bellevue, WA 98004-5973
(425) 455-5575
Fax (425) 455-1046

SPECIFICATION

TITLE OF THE INVENTION

Copy Protection of Portable Game Software

RELATED APPLICATIONS

[0000] This is a continuation-in-part of copending U.S. patent application Serial Number 10/639,038 filed Aug 11, 2003, and a continuation-in-part of Serial Number 10/613,902 filed July 3, 2003 (abandoned), and a continuation-in-part of Serial Number 10/427,793 filed April 30, 2003, and a continuation-in-part of Serial Number 10/135,319 filed April 29, 2002. These applications in their entirety are incorporated by reference herein.

BACKGROUND OF THE INVENTION

Field of the invention

[0001] This invention relates generally to electronic game systems and more particularly to game software distributed in encrypted form for copy protection.

Description of the prior art

[0002] Portable game systems that generate player controlled objects in simulated worlds for display on an LCD screen are well known and are described in US patent 6,369,827. It is also well known to store game program instructions and graphics data in digital memory cartridges that plug into such portable game systems. Even if such digital memory cartridges include a trademark and copyright notice as described in US patent 5,184,830, software pirates disregard such notices. Game software in executable form is easily copied and is often sold by software pirates in counterfeit cartridges and disks and is distributed freely on the Internet. It is also known to protect programs by storing them in a digital

memory in the same processor chip that executes the program instructions as described in US patent 6,339,815. It is also known to include microprocessors in portable game cartridges as described in US patent applications 2002/0028710 and 2003/0050116. Crypto microprocessors that execute encrypted programs using bus encryption are also disclosed in my US patent 4,278,837. It is also known to transmit video game software in encrypted form over a data transmission network.

[0003] Software for game systems has been distributed on laser-readable optical disks for use in game systems. Game software is typically pressed into optical disks during disk fabrication and may be encrypted for copy protection. More than a hundred patents have been issued for optical disks, encryption, and related technologies, such as US patents 6,081,785 and 6,175,629.

[0004] Piracy of portable game software (programs and data) is similar to piracy of music software. When digitized music is read from a data storage medium or decrypted so that it can be converted to analog sounds that can be heard, the digitized music is easy for pirates to copy. But there is one major difference between music and game software. Game programs do not have to be heard or seen by their users and hence game programs do not have to be executed in easily accessible portable game system processors.

[0005] In the present invention, encrypted game programs can be distributed in cartridges and decrypted, stored, and executed in integrated crypto processors in game systems to generate game data, without the game programs being accessible outside of the crypto processors.

SUMMARY OF THE INVENTION

[0006] It is a primary objective of the present invention to provide copy protection for game software that is used in electronic game systems. It is another objective of this invention that this protection be provided at low cost.

[0007] The preferred embodiment of this invention is an electronic game system for distributing game software (programs and data) in encrypted form on an optical disk or in a memory cartridge together with an encrypted key for decrypting the encrypted software. Alternatively, the encrypted software may be downloaded from a server into a memory cartridge or other data storage device together with an encrypted key. Non-encrypted software may accompany the encrypted software. The game system requires a crypto processor that decrypts the encrypted software for execution. Decrypted programs are preferably executed in the crypto processor chip and are not externally accessible, but may also be executed in a conventional processor. Encrypted software on an optical disk is accompanied by a second crypto processor chip containing the encrypted key.

ADVANTAGES

[0008] By distributing game software in encrypted form, proprietary game software can be delivered securely to users without risk that the software will be illegally copied. Encrypted software can be distributed on optical disks, ROM cartridges, or downloaded for a fee.

[0009] **BRIEF DESCRIPTION OF THE DRAWINGS**

Fig. 1 is a general block diagram of one embodiment that includes a disk cartridge connected to a portable game system.

Fig. 2 is a block diagram of one embodiment that includes a disk cartridge with a first crypto processor connected to a second crypto processor in a portable game system.

Fig. 3 is a block diagram of another embodiment that includes additional encryption.

Fig. 4 is a perspective view of a portable game system with a disk cartridge and crypto processor.

Fig. 4a is a block diagram of a portable game system with an internal crypto processor and an external cartridge.

Fig. 5 is a block diagram of crypto functions used in a disk pressing plant.

Fig. 6 is a detailed block diagram of one embodiment of crypto processor 52.

Fig. 7 is a block diagram of another embodiment that reads an encrypted key from an optical disk.

Fig. 8 is a block diagram of another embodiment that reads an encrypted key from a bar code burned into an optical disk.

Fig. 9 is a block diagram of another embodiment that includes a ROM cartridge with a crypto processor.

Fig. 10 is a perspective view of two human game players operating portable game systems having LCD devices that display multiple articulated body parts of player controlled characters.

Fig. 11 is a perspective view of a portable game system with a crypto cartridge and displaying a 3D image of a player character and a non-player character.

Fig. 12 is a perspective view of a cartridge circuit board having crypto processor 303 and ROM 97 attached to the circuit board.

Fig. 13 is an example of a memory map illustrating software stored in ROM 91 in crypto processor 52.

Fig. 13a is an example of a memory map illustrating software stored in ROM 313 in crypto processor 303.

Fig. 14 is a block diagram of an embodiment in which an encrypted disk serial number is sent to a game server that downloads software.

Fig. 15 is a block diagram of an embodiment in which a chip identifier is sent to a game server that downloads software.

Fig. 16 is a memory map of RAM 90 storing game software from an optical disk.

Fig. 16a is a memory map of RAM 132 storing downloaded game enhancements.

Fig. 17 is a block diagram of a game server downloading encrypted game software enhancements.

Fig. 18 is a memory map illustrating one kind of enhancement software having an encrypted address table.

Fig. 19 is a block diagram of a preferred embodiment in which an encrypted chip identifier is sent to a game server that downloads software.

Fig. 20 is a block diagram of crypto communications between the game server and the memory cartridge 16 shown in Fig. 19.

Fig. 21 is a block diagram of a game server downloading encrypted game software to a memory cartridge 16 shown in Fig. 19.

Fig. 22 is a block diagram of a game distribution system in which a retail computer helps transfers data between server 120 and cartridge 16.

Fig. 23 is a block diagram of crypto communications between the game server and the memory cartridge 16 crypto processor chip 303 shown in Fig. 22.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENT

[0010] Fig. 1 illustrates a preferred embodiment of a game system that comprises crypto memory cartridge 16 connected to portable game system 47. The objective is to securely execute and process in portable game system 47 the encrypted software (programs and data) read from tracks 82 of disk 43 housed in cartridge 16. The software on disk 43 (or in semiconductor memory 97 in Fig. 9) is block encrypted using a symmetric digital key K1. Disk 43 is accompanied in each cartridge 16 by a crypto processor 303 (detailed in Fig. 2) that contains key K1 that enables decryption of the encrypted software in crypto processor 52 which is soldered into the circuit board in portable game system 47. Decryption key K1 is not permanently stored in crypto processor 52 or in portable game system 47 and typically would be changed for each game title. When key K1 is changed for encrypting the software on disk 43, key K1 is also changed in crypto processor 303. To prevent accidental separation of disk 43 and corresponding processor 303, both components are housed together in the same cartridge 16.

[0011] Because key K1 is stored and distributed in crypto processor 303 but is used for decryption in crypto processor 52, crypto processor 303 encrypts key K1 and transmits the encrypted key through connectors 247 and 279 and through data bus 93 to crypto processor 52. Crypto processor 52 (detailed in Figures 2, 3, and 6) decrypts the encrypted software read from disk 43 and stores the decrypted software in volatile memory 90 (see Fig. 6) which is on the same chip as processor core 134 in crypto processor 52 which executes the decrypted programs from memory 90. Neither crypto processor 303 nor 52 reveal keys or decrypted program instructions outside of the processor chips.

[0012] Although pirates can make unlimited copies of the encrypted software and perhaps distribute them on the Internet, the encrypted software cannot be decrypted and executed without the two crypto processors 303 and 52 which cannot be copied without knowledge of the inaccessible secret keys.

[0013] Although some of the software for each game may be unencrypted on disk 43 and be loaded into RAM 53 and be executed and processed by conventional processors 50 and 301, this unencrypted software is useless without the decrypted software that is processed by crypto processor 52 to interact with unencrypted software processed by processor 50.

[0014] Decrypted programs executed in crypto processor 52 may generate partially processed game data, such as locations and directions of player controlled objects and points of view. This partially processed game data is further processed by conventional processor 50 and image coprocessor 301 to generate rapidly changing pixel display data in VRAM 302 for display on LCD screen 22.

[0015] Preferably, a small fraction of the programs read from disk 43 would be encrypted and executed by processor 52 while most of the programs from disk 43 would be loaded into RAM 53 and executed by processor 50. Programs that are easy for pirates to reverse engineer need not be encrypted. Programs that are difficult to reverse engineer and do not require rapid access to image co-processor 301 would be suitable for encryption and for execution in crypto processor 52.

[0016] Portable game system 47 may remain an open system for non-crypto cartridges 16 which do not require processor 303 and which store conventional software processed by conventional processors 50 and 301. Prior-art cartridges containing unencrypted programs would not use crypto processor 52.

[0017] Conventional processor 50 is shown in Fig. 1 receiving control data from manual control members such as joystick 20, cross-switch 15, and button-switch 14, but some of the control data should be routed by processor 50 through data bus 93 to crypto processor 52.

[0018] Fig. 2 illustrates an embodiment in greater detail. For clarity, only the encrypted software will be discussed. Encrypted game programs and data are read from tracks 82 on optically readable disk 43 into RAM buffer 97. Accompanying disk 43 is crypto processor 303 containing a non-volatile data memory storing a block of data that includes symmetric key K1 (reference 100), game title identification number 114, and serial number 101.

[0019] Each crypto processor chip 303 in this embodiment has a different serial number that is loaded into the crypto processor chip 303 during assembly of cartridge 16 (see Fig. 5) so that each block of encrypted data on line 61 will be encrypted differently. This prevents pirates from using cryptanalysis that depends on constant unencrypted data such as key K1 and game id for a given game title.

[0020] Block encryption process 147 encrypts the data block (K1, game id, serial) to produce a block of encrypted data on line 61 for transmission to crypto processor 52. The key used for this transmission should not be the same key for every transmission, because this would provide constant encrypted key data that could be distributed on the Internet.

[0021] Instead, a different session key 304 is used every time data is transmitted on line 61 to processor 52. This session key 304 is generated by random number generator 311 in crypto processor 52. To prevent a pirate from supplying an unauthorized encrypted K1 key on line 61 or supplying a bogus session key, the session key 304 is encrypted by block encryption process 306 under control of a symmetrical chip key 131 (key K3).

[0022] Crypto processor 303 has the same chip key K3 so that only genuine processor chips 303 can decrypt the encrypted session key in block decryption process 307. Since random number generator 311 in processor 52 generates a different session key for each transmission to processor 303, the encrypted session key is also different every time process 306 transmits an encrypted session key to processor 303.

[0023] This deprives pirates of a constant session key they need for cryptanalysis. It also prevents pirates from bypassing processor 303 by sending unauthorized data directly into processor 52 on lines 61 and 71. Lines 61 and 71 are part of data bus 93 in Fig. 1 and may be multiplexed.

[0024] To insure that session key 304 is truly random and not pseudo random, a thermal noise source 310 can generate seed for random number generator 311 as described in US patent 4,694,412.

[0025] When the encrypted session key is decrypted by process 307 under control of chip key K3, the resulting plain session key 304 in processor 303 controls block encryption process 147 of key K1, game id, and serial for transmission on line 61 to processor 52.

[0026] When the encrypted block (key K1, game id, serial) is received on line 61 by crypto processor 52, it is block decrypted using the same random session key generated microseconds earlier by random number generator 311.

[0027] Decrypted key K1 is then used by block decryption process 111 to decrypt blocks of encrypted programs and data from RAM 97 to produce decrypted blocks of programs and data that are stored into RAM 90.

[0028] One of the first blocks to be decrypted by process 111 may include the game title identification number in RAM 90 which is then compared to the decrypted game title identification number 114 by verification process 136. If the two game id's do not match, an error message may be displayed on LCD 22.

[0029] Decrypted serial number 101 in crypto processor 52 should not be revealed outside of processor 52 because that would provide pirates with known plaintext to encryption process 147. However, a block encrypted serial number (not shown in Fig. 2) may be displayed on LCD 22.

[0030] Fig. 3 illustrates another embodiment in which the key data block (key K1, game id, and serial) is stored in crypto processor 303 in encrypted form in non-volatile memory 94. Encrypted key K1 is preferably doubly encrypted by block encryption process 147 in processor 303 under control of session key 304. The encrypted data block in memory 94 is previously block encrypted under control of key K2 that is not stored in processor chip 303. This is to deter insider theft of key K1 during manufacturing. Crypto processor 52 doubly decrypts key K1, game id, and serial using session key 304 and chip key 98 (K2).

[0031] Workers in plants that load key data into crypto processor 303 would not know the values of key K2. Workers in plants that load key data into crypto processor 52 would not know the values of key K1. This separation of functions is important for internal security.

[0032] Fig. 4 is a perspective view of portable game system 47 with disk cartridge 16 containing optical disk 43 and crypto processor 303. Serial port 40 is for a link cable to connect portable game system 47 to a console video game system so that portable game system 47 may be used in place of a controller or as an auxiliary display.

[0033] Fig. 4a is a simplified block diagram illustrating portable game system 47 with an internal crypto security processor 52, external memory cartridge 16, and serial port 40.

[0034] Fig. 5 is a block diagram illustrating disk fabrication processes used in a disk pressing plant for writing data onto disk 43 and into crypto processor 303. Random number generator 55 generates a pseudo random symmetrical encryption/decryption key 100 (key K1). Key 100 controls block encryption process 133 which encrypts plain game programs and data 104 to produce encrypted game programs/data 97 which are molded as tracks 82 into disk 43 by disk molding/pressing machine 149.

[0035] Key 100 (key K1), game title identifier 114, serial number 101, and other optional encrypted keys are block encrypted by encryption process 129 under control of key 98 (key K2) selected from key table 110 to produce encrypted block 94 which is stored into crypto processor 303 in the Fig. 3 embodiment. A randomly generated key selection number 113 specifies which key 98 is selected from table 110.

[0036] Adder 64 generates a disk identification serial number 101 which is different for each disk. Key 98 (key K2) is also stored into crypto processor 52.

[0037] Alternatively, the key block may be molded as track 148 into disk 43 (see Fig. 7) by disk molding/pressing machine 149 at the same time tracks 82 are pressed.

[0038] In addition, key selection number 113 (see Fig. 6) may be stored into crypto processor 303 or disk 43.

[0039] Fig. 6 is a detailed block diagram of one embodiment of crypto processor 52. To deter pirates from providing bogus encrypted blocks on line 61, a random bit string is generated by crypto processor 52 and sent to crypto processor 303 which processor 303 immediately alters and returns to processor 52 on line 61. Response timer 314 in processor 52 measures the number of clock cycles between sending the bit stream and receiving the correct response in processor 52. Responses delayed less than m clock cycles or more than n clock cycles are rejected as bogus.

[0040] When decrypted programs and data are stored into RAM 90, processor core 134 executes decrypted program instructions from RAM 134 in addition to instructions stored in boot ROM 91.

[0041] Crypto processor 52 typically executes decrypted programs from memory 90 while cartridge 16 is inserted into portable game system 47. Alternatively, cartridge 16 may be removed from connector 279 and the decrypted game programs in memory 90 may be executed by processor 52 as long as portable game system 47 is electrically powered.

[0042] Communication between crypto processor 52 and conventional processor 50 is through SRAM 239 that is bus multiplexed as described in detail in my copending US patent application serial number 10/427,793 filed April 30, 2003.

[0043] Fig. 7 is a block diagram of another embodiment in which encrypted key K1 is read from track 148 on disk 43 instead of being stored in crypto processor 303. Game identifier 54 is stored into crypto processor 303 for comparison with encrypted game identifier 114 in crypto processor 52 to prevent pirates from using one processor 303 for many different games.

[0044] Fig. 8 is a block diagram of another embodiment in which encrypted key K1, game id, and serial number are read from bar code 80 that is laser burned into optical disk 43 after molding instead of being stored in crypto processor 303. Game identifier 54 is stored into crypto processor 303 for comparison with encrypted game identifier 114 in crypto processor 52 to prevent pirates from using one processor 303 for many different games.

[0045] Fig. 9 is a block diagram of another embodiment in which encrypted programs and data are read from semiconductor memory 97 instead of from optical disk 43. Memory 97 and crypto processor 303 are housed in cartridge 16 (see Fig. 12).

[0046] Fig. 10 is a pictorial view of two human game players 10 and 12 operating portable game systems 44 and 47 having LCD devices that display multiple articulated body parts of player controlled characters, such as arm 59, hand 36, and wrist 37. Player controlled objects are exemplified by pipe 35.

[0047] Fig 11 is a perspective view of portable game system 47 displaying a 3D picture on LCD 22 of two animated characters that have multiple body parts. Movement of the body, arms, and legs of the human-like player character are controlled by manually operated control devices such as cross-switch 15, push button switches 14, and other manually operated devices. One embodiment of cartridge 16 is shown in Fig. 11 inserted into portable game system 47.

[0048] Fig 12 is a perspective view of a cartridge circuit board 299 with crypto processor 303 and ROM 97 attached. ROM 97 can be a separate chip as shown, or it can be included on the crypto processor 303 chip. See Fig. 9 for this embodiment.

[0049] Fig. 13 is an example of a memory map of program instructions and data stored in boot ROM 91 in crypto processor 52 for execution and processing by processor core 134. Some of these instructions may be stored in RAM 90 instead. For example, some of these instructions may copy position data, location data, direction data, and/or texture data from RAM 90 to SRAM 239.

[0050] Fig. 13a is an example of a memory map of program instructions and data stored in boot ROM 313 (not shown) in crypto processor 303.

[0051] Fig. 14 illustrates an embodiment of a video game system 19 that includes crypto processor 52 (preferably attached to the motherboard of video game system 42), memory cartridge 16 containing downloaded software, Internet 121 or telephone line data transmission, optical disk 43, TV 11, and components housed in video game console unit 42. For clarity, the housing of unit 42 is not shown. Disk reader 83 reads video game programs and data from tracks 82 into RAM 90. Disk reader 83 may also read game identifier 114 from lead-in track 148. Tracks 82 and 148 are typically pressed into disk 43 during disk fabrication. Disk reader 83 may also read encrypted control record 94 from bar code 80 which is typically burned into optical disks in a ring-shaped "Burst Cutting Area" (BCA) after the pressing process. Each bar code 80 includes an encrypted control record which includes a unique serial number 101 for each disk (see Figures 4, 5, and 6). Serial number 101 is block encrypted together with authenticating data or random filler bits.

[0052] Game software in RAM 90 consists of programs and data read from disk 43. These programs execute in processor 86 which may include one or more coprocessors. Execution is typically from on-chip cache memory 128 which is faster than executing from RAM 90. Programs executed in processor 86 process data from RAM 90 and generate picture data from which video signal generator 117 generates video signals for display on TV 11 or other video display such as LCD flat panel displays. Processor 86 may also be connected to one or more portable game systems 44 or other user input control devices by cables or wireless equivalent (not shown in Fig. 14) such as infrared, ultrasonic, RF waves, or other data communicating forms of energy.

[0053] When enhancement software is downloaded from game server 120, an Internet or telephone line connection 121 is temporarily established between server 120 and a modem located in console 42 or attached to console 42. For clarity in Fig. 14, this modem is illustrated as a sending (modulating) modem 137 and a receiving (demodulating) modem 138 separately to show data flow. Game identifier 114 read from disk 43 and encrypted control record 94 read from disk bar code 80 are transmitted to game server 120. Game identifier 114 selects enhancement software for downloading and encrypted control record 94 is decrypted in the server for customer identification (see description below with reference to Fig. 6).

[0054] In the preferred embodiment, server 120 downloads at least four kinds of enhancement software: non-encrypted programs and/or data 96, encrypted programs and/or data 97, key selection byte 113, and encrypted key 124 (K1). These four kinds of downloaded software, and game identifier 114, and encrypted control record 94 are stored in removable nonvolatile read/write data memory cartridge 16 or harddisk (not shown). It is recommended that users download enhancements into a separate cartridge 16 for each disk 43 to insure there will be sufficient memory space for future enhancements. When remaining cartridge memory space is insufficient for further enhancements, it is time for an upgrade to the next edition of disk 43 in the game series.

[0055] Non-encrypted downloaded software 96 is copied into RAM 132 (RAM B) for execution and/or processing by processor 86 for additional levels and other enhancements. Software in RAM 132 would typically have substantially fewer bytes than software in RAM 90 because enhancement software 96 in RAM 132 makes use of unfinished and unplayable

software read from disk 43 into RAM 90. RAM 90 and RAM 132 may be different parts of the same RAM memory, but are shown separately in Fig. 14 for clarity.

[0056] Encrypted software 97 is also downloaded from server 120 and provides essential but secret functions for enhancement software in RAM 132 to execute. By keeping secret a small but essential part of the enhancement software, protection against illegal software copying is provided. For example, digital links between software in RAM 132 and RAM 90 would typically be required. If these links are in RAM 132 or RAM 90, they are at risk of being illegally copied. But if these digital links are never stored in RAM 132 or 90 in decrypted form, but instead are stored in inaccessible SRAM 104, the links cannot be copied. Another example is a small program decrypted in decryption box 111 in crypto processor 52 and stored in SRAM 104 and executed in processor core 134. This small program may provide a trade-secret method of character movement, character intelligence, special sound generation, or other game element that is difficult to program and therefore may provide a competitive advantage if securely executed in crypto processor 52, rather than in processor 86.

[0057] Processor core 134, RAM 104, buses 115 and 116, and other components in crypto processor 52 should be integrated into one application-specific circuit that is physically protected as described in my bus-encryption patent US 4,278,837. Secret programs, data, and keys are stored in volatile static RAM powered by electric battery or cell 130. Attempts to physically probe, scan, or peel the crypto chip should result in loss of voltage to RAM 104 and key table 110, which should result in destruction of all secret data in crypto processor 52.

[0058] Briefly referring to Fig. 17, enhancement software 97 is encrypted in server 120 using block encryption method 133 such as DES or a similar block encryption method under control of a secret session key 100 (key K1) which is a random number generated by server 120. Key K1 is then encrypted to produce encrypted key 124 (encrypted key K1) using a block encryption method 129 (which may be the same or similar to 133 and 99) under control of secret key K2 selected from secret key table 110 by a one-byte key selection number 113. Key selection number 113, and encrypted key 124 (encrypted key K1) are downloaded, along with encrypted software 97, and non-encrypted software 96 to game console 42.

[0059] With reference to Fig. 14, each crypto processor 52 has the same table 110 of secret keys mentioned above with reference to Fig. 19. Key selection number 113, hidden somewhere in encrypted software 97, selects key K2 from key table 110. Key K2 then controls block decryption 99 to decrypt encrypted key 124 (encrypted key K1) to produce plain decryption key K1 in register 100. Key K1 then controls block decryption 111 of encrypted software 97 which is read from memory cartridge 16 by crypto processor 52, one block at a time, into input buffer 103, to produce decrypted blocks on internal data bus 115. After performing cyclic redundancy check (CRC) 136 on each decrypted block, processor core 134 stores each block of decrypted program and data into battery-powered SRAM 104.

[0060] Although it may be tempting to simplify crypto processor 52 by eliminating key table 110, this would result in one secret master key K2 that would be the same in every crypto processor chip 52. That would be excessively risky and is not recommended. The keys in key table 110 can be further disguised by mixing randomly located decoy bits among the key bits in table 110. If an intruder succeeded in penetrating the physical

security of crypto chip 52 without loss of secret data in SRAM 104 and key table 110 and discovered the bit values in table 110, the intruder would not be able to make use of those bits as key without also discovering the program instructions in ROM 134 and SRAM 104 and learning where to remove the decoy bits from key K2 read from table 110.

[0061] After decrypted programs and data are stored in SRAM 104, processor core 134 executes the programs in SRAM 104 which communicate with processor 86 by a series of digital semaphores in input buffer 103 and output buffer 105. Using semaphores, perhaps encrypted, avoids the possibility of an intruder addressing data in SRAM 104. Internal address bus 116 and data bus 155 should be inaccessible, either as input or output, from outside of crypto chip 52. Data in RAM 132 and/or RAM 90 needed by processor core 134 is indirectly passed by way of processor 86 and buffer 103. Data needed by processor 86 to link software in RAM 132 and RAM 90 need not be stored in either RAM 132 or RAM 90. Instead, the linkage data may be read into cache 128 which should be inaccessible by probing or at external terminals.

[0062] With reference to Fig. 15, most of the components and functions discussed above with reference to Fig. 14 are the same in Fig. 15, except for the following differences: The encrypted control record 94 that is read from bar code 80 on disk 43 and which is different for every disk, is not available to memory cartridge 16 in this example. Hence, encrypted control record 94 is not available for transmission to server 120 in Fig. 15. To prevent pirates from distributing unauthorized copies of encrypted programs/data 97 and encrypted key 124, each crypto processor chip 52 in Fig. 15 includes a unique, inaccessible, chip identifier 139 which is different for every crypto chip 52. This chip identifier 139 is transmitted to game server 120 to identify the customer's game system. Game server 120

retransmits the chip identifier back to modem 138 in encrypted data block 97 or block 124 so that processor core 134 can compare the returned chip identifier to hardware chip identifier 139 in crypto processor chip 52. If there is no match, further decryption of encrypted programs 97 is inhibited. Chip identifier 139 may also be used in the Fig. 14 system so that records may be kept in server 120 identifying which disk serial numbers are used with which crypto processor chip 52.

[0063] Fig. 16 is a memory map of RAM 90 (RAM A) for exemplary software read from disk 43. These programs and data are typical for conventional video games. Fig. 16a is a memory map of RAM 132 (RAM B) for exemplary non-encrypted enhancement software 96 downloaded from server 120. Software 96 would typically not include all software needed to play an enhancement level, but after downloading would be linked to software in RAM A read from disk 43. Such a combination of software in RAM A and RAM B would provide new playable game levels.

[0064] Fig. 17 illustrates functions of game server 120 that provides downloadable game enhancements. Game enhancement software is stored in database 122 together with corresponding key selection number 109, and key selection number 113. Customer records for each game title are stored in database 143 by disk serial number 101 or chip identifier 139. When an enhancement is requested over an Internet or telephone link 121, game identifier 114 and encrypted control record 94 are uploaded to server 120, as described above with reference to Fig. 14.

[0065] Using game identifier 114, database reader 123 reads data from database 122 to get the corresponding key selection number 109. Selection number 109 selects key K4 from key table 110. Encrypted control record 94, that was read from bar code 80 on disk 43 in Fig. 14, is then decrypted by block decryption process 142 under control of key K4 to produce decrypted serial number 101 and random filler bits. Process 102 checks serial number 101 against database 143 to determine which enhancement number 141 is next for downloading. Database reader 123 reads enhancement software from database 122 indicated by enhancement number 141 and game identifier 114. Non-encrypted software 96 is transmitted to a user's video game console 42 as described above with reference to Fig. 14.

[0066] Before enhancement software 97 is downloaded, it is block encrypted by process 133 under control of a randomly generated decryption key 100 (key K1). Key selection number 113 selects key K2 from key table 110 and block encryption process 129 under control of key K2 encrypts key K1 to produce encrypted key 124 which is downloaded to the user's console 42.

[0067] Fig. 18 is a memory map that illustrates digital links (1, 2, 3, 4, 5, 6) between portions of non-encrypted disk software stored in RAM 90 (RAM A) and downloaded portions of non-encrypted software 96 stored in RAM 132 (RAM B). Downloaded address table 140 of these digital links is reencrypted by a program in SRAM 104 (Fig. 14) and is sent from crypto processor 52 in reencrypted form on data bus 93 to processor 86 which decrypts table 140 and stores it in cache 128. The combined software in RAM A and RAM B is not executable without decrypted table 140 in cache 128. Address table 140 in decrypted form is not stored in RAM 90 or RAM 132, nor is it accessible from processor 86

or data bus 93. Encrypted address table 140 is decrypted in processor 86 using program instructions already in cache 128 and/or ROM in processor 86.

[0068] As described above with reference to Fig. 14, several other trade secret programs may be securely decrypted and stored in SRAM 104 in crypto chip 52 and executed in processor core 134. This may provide a proprietary method of character movement, character intelligence, special sound generation, or other game element that is difficult to reverse engineer and program and therefore may provide a competitive advantage if securely executed in crypto processor 52.

[0069] Fig. 19 illustrates a preferred embodiment of a video game system 19 that includes crypto processor 52, memory cartridge 16, Internet 121 or telephone line data transmission, modem 137/138, TV 11, and components housed in video game console unit 42. Crypto processor 52 should be attached to the motherboard of video game console 42 or portable game system 47 so that attempts to replace processor 52 with an unauthorized processor would destroy traces within the motherboard. For clarity, the housing of unit 42 is not shown.

[0070] Unlike the examples in Figures 1-8, 14 and 15, the example shown in Fig. 19 has no optical disk 43 or crypto processor 303. Like the unique chip identifier 139 in Fig. 14, a unique chip identifier 139 is stored in crypto chip 52 in Fig. 19. Chip identifier 139 is sent to game server 120 in encrypted form to deter known-plaintext attacks on key block 94. Encryption process 147 block-encrypts chip identifier 139 together with a random serial number using a random session key 304 (K4) to produce encrypted block 323 (see Fig. 20). Encrypted block 323 is transmitted to game server 120 along with requested game identifier 114.

[0071] Server 120 downloads the requested non-encrypted software 96 into cartridge 16 and block encrypts the requested encrypted software 97 as a function of randomly generated key K1 (see Fig. 21). Server 120 also block encrypts key K1 together with chip identifier 139 and random filler bits to produce key block 94. The encrypted key block 94 is also downloaded from server 120 into cartridge 16. In crypto processor 52, block decryption process 99 decrypts key block 94 to produce decrypted key K1 (reference 100), decrypted chip identifier 139, and filler bits (not used). It is important that these data fields be encrypted together as one block and not as individual fields or bytes, so that each bit in the encrypted block 94 is a complex function of every bit of the decrypted block and of every bit of key K2.

[0072] After key block 94 has been decrypted by process 99, verification process 136 compares decrypted chip identifier 139 to hardware chip identifier 139. If they are different, decryption of programs/data 97 by process 111 is inhibited. The random filler bits in encrypted key block 94 should be at least 64 bits to insure that chip identifier 139, if discovered, cannot be used in a known-plaintext attack.

[0073] Downloaded non-encrypted programs 96 are executed in processor 86 which may include one or more coprocessors. Programs executed in processor 86 process data from RAM 96 and generate picture data in RAM 90 from which video signal generator 117 generates video signals for display on TV 11 or other video display such as LCD flat panel displays. Processor 86 may also be connected to one or more portable game systems 44, 47, or other user input control devices by cables or wireless equivalent (not shown in Fig. 19) such as infrared, ultrasonic, RF waves, or other data communicating forms of energy.

[0074] Fig. 20 illustrates crypto communication between game server 120 and crypto processor chip 52 in greater detail than shown in Fig. 19 and Fig. 21. In the right and lower portion of Fig. 20, decryption of key block 94 and encrypted game programs and data 97 are shown. These functions are also described above with reference to Fig 9. Both key block 94 and encrypted programs 97 are downloaded from server 120 and decrypted in crypto processor 52 as shown in Fig. 19. To prevent pirates from distributing this encrypted data 94 and 97 in bogus cartridges and from the Internet, it is necessary in this example for the downloaded data block 94 to be usable only in the specific crypto processor 52 owned by a person who paid for a license to use encrypted game programs and data 97.

[0075] Key block 94 is made different for each user by including a unique chip identifier 139 in each crypto processor chip 52 and in each block encryption process 129 in the game vendor's server 120. Chip identifier 139 is a unique, inaccessible, and unalterable binary number in each crypto processor 52. Chip identifier 139 is shown in Fig. 15 being transmitted to server 120 in the clear and in Fig. 19 being encrypted in process 147 before being transmitted to server 120. This encryption process is shown in detail in Fig. 20 and prevents pirates from using chip identifiers 139 with corresponding key blocks 94 as plaintext/ciphertext pairs for cryptanalysis.

[0076] Chip identifier 139 in crypto processor 52 in Fig. 20 is encrypted together with random filler bits (not shown) by block encryption process 147 to produce encrypted chip identifier 323 under control of a session key 304 (K4) that is randomly generated by random number generator 311 in server 120. Session key 304 is first generated in server 120, encrypted by process 306 under control of key 131 (K3) and transmitted in encrypted form to decryption process 307 to produce a plain session key 304 in crypto processor 52. This

session key process is also described above with reference to Fig. 2. Encryption of session keys prevent their use in cryptanalysis of encrypted chip identifiers 139. Symmetric keys 131 are shown for encryption methods such as DES, but nonsymmetric public key / private key pairs may also be used for processes 306 and 307.

[0077] After encrypted chip identifier 139 is decrypted in block decryption process 142 in server 120, plain chip identifier 139 is encrypted together with key 100 (K1) and random filler bits by block encryption process 129 under control of key K2 (reference 98 from key table 110) as described above with reference to Fig. 17 and below with reference to Fig. 21. This encryption process 129 produces key block 94 which is downloaded by server 120 to RAM in cartridge 16 along with encrypted programs/data 97 as described above with reference to Fig. 19.

[0078] When cartridge 16 is inserted into a socket 279 in video game system 42 (Fig. 19) or portable game system 44 or 47 (Fig. 1), encrypted key block 94 is copied from cartridge 16 into input buffer 103 in crypto processor 52 (Fig. 19). Encrypted key block 94 is then decrypted in block decryption process 99 under control of key 98 (K2) to produce a decrypted block comprising key 100 (K1), chip identifier 139, and filler bits (not used).

[0079] Decrypted chip identifier 139 is then compared to hardware chip identifier 139 to determine if they match. If they do not match, decryption of encrypted programs 97 is inhibited and a message is displayed on TV 11 or LCD 22 that game programs in memory cartridge 16 cannot be used with this hardware system 42 or 44 or 47. The cartridge 16 hardware can be used with any game system designed for it, but cartridge 16 should also contain matching software 94 and 97 to be usable, in this example.

[0080] Fig. 21 illustrates functions of game server 120 that provides downloadable game software. Game software is stored in database 122 together with corresponding key selection number 113 which may be different for each game title. Customer records for each game title purchased are also stored in database 143 by chip identifier 139. When game software is requested over an Internet or telephone link 121, requested game identifier 114 and encrypted chip identifier block 323 are uploaded to server 120, as described above with reference to Figures 19 and 20.

[0081] Database reader 123 reads requested game software from database 122 specified by game identifier 114. Non-encrypted software 96 is transmitted to a user's video game console 42 as described above with reference to Fig. 19.

[0082] Encrypted chip identifier block 323 contains chip identifier 139 encrypted together with a random filler bits. Block 323 is decrypted in server 120 using block decryption process 142 such as DES or a similar block encryption method under control of session key 304 (K4) generated by random number generator 311. Decryption of block 323 produces decrypted filler bits 101 (not used) and decrypted chip identifier 139. The function of random filler 101 is to deter cryptanalytic attacks on block 323. Decrypted chip identifier 139 is checked for validity by process 102 by lookup in database 143 in addition to CRC validation.

[0083] Encrypted software 97 is encrypted in server 120 using block encryption method 133 such as DES or a similar block encryption method under control of a secret decryption key 100 (key K1) which is a random number generated by server 120.

[0084] Random decryption key 100 (K1), chip identifier 139, and more random filler bits are then block encrypted to produce encrypted key block 94 using a block encryption process 129 (which may be the same or similar to 133 and 147) under control of secret key K2 selected from secret key table 110 by key selection number 113. Key selection number 113, and encrypted key block 94 are downloaded, along with encrypted software 97, and non-encrypted software 96 to game console 42 which stores them into cartridge 16.

[0085] Fig. 22 illustrates an embodiment of a video game system 19 that comprises components shown in Fig. 19 and also has an additional computer 324 operated by a retailer to transfer data between server 120 and memory cartridge 16. The user of video game console system 42 or portable game system 47 or 44 will take cartridge 16 to the retailer to buy game software downloaded from server 120 through the retailer's computer 324. Retailer computer 324 has software to access server 120 through the Internet 121 and/or telephone connection and to request purchase of game software specified by game identifier 114. Retail computer 324 also has a socket (not shown) into which cartridge 16 is inserted and electronic components (not shown) to store downloaded software through the socket into cartridge 16 memory which may be EEPROM, battery-powered SRAM and/or other data non-volatile storage media, as shown for example in Figures 1, 9, 15, and 19. Software in retail computer 324 would also include accounting programs to record monetary amounts payable to the owner of game software being downloaded from server 120.

[0086] Cartridge 16 has a second crypto processor 303 (described above with reference to Fig. 2 and further described below with reference to Fig. 23) that transfers the value of chip identifier 139 to server 120 through retailer computer 324 using randomly generated session keys 304 to deter cryptanalysis. As described below with reference to Fig. 23, server

process 311 generates a random session key 304 (K4) to control block decrypting process 142 which decrypts encrypted chip identifier 323. In Fig. 22 the unique binary value of chip identifier 139 in game system crypto processor 52 is the same value in cartridge crypto processor 303. Hence cartridge 16 crypto processor 303 and crypto processor 52 are a matched set that will operate correctly only if used together.

[0087] Block encryption process 147 in cartridge crypto processor 303 encrypts chip identifier 139 and transmits the encrypted chip id value through retailer computer 324 and the Internet 121 to server 120 which generates encrypted key block 94 as described above with reference to Fig. 20. Server 120 then downloads encrypted key block 94 through the Internet 121 and retailer computer 324 which stores encrypted key block 94 into cartridge 16 as shown in Figures 20 through 23. Server 120 also transmits encrypted software 97 and non-encrypted software 96 and 113 to retailer computer 324 which stores it into cartridge 16 memory. The user of cartridge 16 then inserts it into his game system and the data in cartridge 16 is decrypted by crypto processor 52 as described above with reference to Fig. 19.

[0088] Alternatively, retailer computer 324 may write downloaded encrypted game data onto a writable disk that can be read by a user's game system disk reader 83 as shown in Figures 14-15.

[0089] There is no need for retail computer 324 or cartridge 16 to be secure from tampering because encrypted program data 97 and encrypted key block 94 and encrypted chip identifier 323 pass through retailer computer 324 without change. Even if multiple copies were made of the software in cartridge 16, the programs and data would be usable

only in the game system that contains crypto processor 52 with a corresponding chip identifier 139. If any alterations or substitutions were made to the encrypted data processed by retail computer 324, the data would be unusable.

[0090] Fig. 23 illustrates in greater detail crypto communications between game server 120 and crypto processor chip 303 in memory cartridge 16. Fig. 23 repeats some of the processing shown in Fig. 21 but shows in greater detail how chip identifier 139 is transmitted in encrypted form from crypto processor 303 to game server 120 using a random session key. After random number generator 311 generates session key 304 for decryption process 142 as shown in both Figures 21 and 23, the same session key 304 is further encrypted by block encryption process 306 under control of key 131 (K3). Encrypted session key 306 is transferred through Internet 121 and retailer computer 324 to crypto processor 303 which decrypts the encrypted session key in process 307 under control of the same key 131 (K3) which is permanently stored in processor chip 303. Decryption process 307 produces the plain decrypted session key 304 (K4) which controls block encryption process 147.

[0091] Encryption process 147 encrypts chip identifier 139 together with randomly generated filler bits so that the block being encrypted is a full block (preferably 128 bits). Encrypted chip identifier 323 is transferred from crypto processor chip 303 through retailer computer 324 to server 120 where encrypted chip identifier 323 is decrypted by block decryption process 142 under control of the same session key 304 mentioned above. Server

process 129 then reencrypts chip identifier 139 together with key 100 (K1) to produce an encrypted block which server 120 transfers through Internet 121 and retail computer 324 which stores the encrypted key K1, chip identifier 139, and filler bits into non-volatile memory 94 in cartridge 16.

[0092] In the examples described herein, block encryption and decryption used in processes 99, 111, 142, 147, 306, 307 operate on blocks of at least 64 bits under control of symmetrical keys of at least 64 bits and preferably 128 bits. Such block encryption may use block encryption methods such as the Data Encryption Standard (DES), AES, or similar methods, so that changing any one bit of plaintext affects all bits of ciphertext in an encrypted block, changing any one bit of ciphertext affects all bits of plaintext in a decrypted block, and changing any one bit of an encryption key affects all bits of plaintext in the block, without providing clues that would lead to discovery of the bit values of the secret key through chosen ciphertext attack, chosen encrypted key attack, toggling of bits in ciphertext or encrypted key, differential cryptanalysis, differential fault analysis, and other cryptanalysis techniques. DES is described in detail in Federal Information Processing Standard (FIPS) 46, Nov 23, 1977; and in FIPS PUB 46-3, October 25, 1999.

[0093] DES is considered obsolete because it has been successfully cracked using differential cryptanalysis with massive amounts of plaintext-ciphertext pairs. But in the present invention, there need not be any plaintext-ciphertext pairs. The decrypted programs are stored in RAM 90 and are not revealed outside of crypto processor 52. Likewise much of the data goes no farther than RAM 90 and processor core 134. By encrypting only program instructions and literal data in instructions, but leaving unencrypted the data that is transferred on bus 93 to processor 50 in Fig. 1, there will be no plaintext-ciphertext pairs that a pirate could use in a cryptanalysis attack. Without known plaintext, DES is more than adequate for this application.

[0094] Simplified variations of DES may therefore be used for the present block encryption/decryption processes. The initial permutation IP and the inverse permutation IP^{-1} can be removed because they were apparently designed for use with ASCII text messages. Executable program instructions are not ASCII and are not text messages. The DES key schedule can also be removed because it was designed to limit the key to 56 bits, a limitation that was subsequently relaxed.

[0095] Although encrypted data is accessible on bus 61, encryption of variable session keys prevents access to encrypted-unencrypted pairs. Hence keys K1, K2, and K3 would be very difficult to discover.

[0096] To distinguish encrypted data from non-encrypted data on disk 43, a table can be recorded on disk 43 with tags to indicate which address ranges are reserved for encrypted or non-encrypted data. Tags could also be recorded in header data that precede encrypted data and precede non-encrypted data.

[0097] Symmetric key block encryption uses the same secret key for decryption and for encryption. Typically this key is at least 64 bits and preferably 128 bits or larger. In the preferred embodiment, there is not one master key in processors 303 or 52, because if it were compromised, perhaps by an employee or contractor of the game vendor, the processors would become useless. Instead, in the preferred embodiment, each of crypto processors 303 and 52 includes key table 110 (see Fig. 6) so that secret key K2 and K3 can be changed in mid production of any game title by changing to a different key in the table. If the key bits in table 110 are intermingled with unused random decoy bits, anybody who accesses the bits will not know which bits are key bits without also reading the on-chip

ROM or RAM program that access bits that are key among bits that are decoys and reconstruct their sequence.

[0098] Key table 110 in processors 303 and 52 may be stored in an SRAM powered by a battery 130, so that attempts to probe, scan, or peel processor chips 303 or 126 would break a power trace and destroy the keys in table 110. If key table 110 were mask programmed or stored in EEPROM or flash ROM, that would reduce security of the keys, unless the key bits were rearranged and/or distributed among decoy bits. Keys should not be externally readable or changeable in crypto processors 303 or 52. Key table 110 should be physically protected against probing, chip peeling, scanning electron microscopy, and voltage-contrast imaging. Physical security for chip keys is described in detail in my US patent 4,278,837 for crypto microprocessors that use bus encryption.

[0099] Processor core 134, includes an ALU, registers, a stack, instruction decoder, and a program counter to address each executable instruction in sequence in ROM 91 and RAM 90, fetch each instruction, and increment the program counter to address the location of the next instruction.

[0100] Crypto processor 52 in this example, executes decrypted programs stored in RAM 90 that generate intermediate game data that may represent variable characteristics of one or more player controlled objects or characters, and/or non-player objects that move across a background, and/or 2D or 3D views of a simulated world. The game data generated in processor 52 may represent positions, locations, and directions of player controlled game objects such as characters with articulated arms and legs and predefined textures. Even if animation of arms and legs is performed by image coprocessor 301, the spatial coordinates,

orientation, and direction of movement of the character may be specified by processor 52 executing the decrypted program instructions in RAM 90.

[0101] The game data generated in crypto processor 52 may also represent positions, locations, and directions of points of view, and may also represent game scores, game status, maps, statistics, object selection, icons, verbal descriptions, instructions, menus, other displayable data, and/or signals to trigger music, voice sounds, and sound effects.

[0102] Data representing background scenery in 2D portable game systems may be unencrypted on disk 43 and loaded into RAM 53 because backgrounds are easily readable or easily reconstructed by pirates. But the program instructions that determine when and what backgrounds are needed and what changes are made to backgrounds (such as a door remaining open) may be executed by crypto processor 52 from RAM 90.

[0103] Image coprocessor 301 in 2D systems may perform scrolling, flipping, blending, scaling, rotating, fading, windowing, and other image processing functions on display data stored in display RAM (VRAM) 302 for display on LCD screen 22. In 3D systems, image coprocessor 301 may perform coordinate transformations of polygons, texture rendering, bump mapping, lighting and shadows, and rasterizing polygon data into displayable pixel data in VRAM 302 for display on LCD 22.

[0104] As used herein, the term "video screen" includes the display area of a television screen, computer monitor, video monitor, RGB monitor, CRT, and the like. The term "video" includes composite, non-composite, RGB, monochrome, color, analog, digital, raster-scanned, MPEG video, and the like.

[0105] The details of cartridge 16 and crypto processors 303 and 52 are given here only as examples and numerous other designs may be used.

[0106] As used herein, the term "molded" includes injection molded, pressed, stamped, and other disk fabrication methods.

[0107] The term "LCD" (liquid crystal display) has been used herein as an illustrative example of any display apparatus having discrete dot-matrix picture elements.

[0108] The term "program" as used herein may consist of more than one loadable module and typically includes executable instruction data and any data that is typically part of a program module or modules.

[0109] The processes of encryption and decryption specified herein may be performed by software, i.e. program instructions executed in a processor core with data tables such as the S-boxes (substitution boxes) that are used in DES. Or the encryption and decryption may be performed in dedicated crypto hardware in the processor chips, or a combination of hardware and software in the processor chips. Encryption and decryption processes may be symmetric (for example DES) or non-symmetric (for example RSA) or a combination of symmetric and non-symmetric.

[0110] Although I have described my invention with a degree of particularity in connection with what is presently considered to be the most practical and preferred embodiments, the present disclosure has been made only by way of illustration and example and is not to be interpreted as restrictive or limiting as to the meaning of words in the patent or its claims. It is understood that various modifications, variations, arrangements, and/or equivalents, can be devised without departing from the spirit and scope of the invention which is defined by the claims.

[0111] Reference Numbers in Drawings

10 human game player
11 television (TV) set or video monitor
12 human game player
14 control switch
15 manual cross-shaped control switch
16 memory cartridge
19 video game system generally
20 joystick
22 LCD screen
26 process of stop reading disk
33 LCD pictures
34 LCD pictures
35 player controlled object
36 simulated hand & arm
37 simulated hand & arm
40 serial link port in portable system
42 video game system console
43 optical disk
44 portable game system
45 cable from controller to console
47 portable game system
49 cursor
50 processor in portable system
52 crypto processor
53 RAM in portable system
54 game id
55 program process
56 video screen
59 cursor
61 data bus connecting crypto processors
64 program process
71 data bus connecting crypto processor
76 boot ROM in portable system
80 burst cutting area (BCA) of disk

81 program and data area of disk
82 tracks molded into disk
83 optical disk reader
84 security processor
86 CPU processor in console
90 SRAM in crypto processor
91 boot ROM in crypto processor
92 address bus
93 data bus
94 encrypted block (key K1, game id, serial num)
95 "public" key
96 non-encrypted programs and/or data in RAM
97 encrypted programs and/or data
98 secret key (K2)
99 process of block decryption (K2)
100 secret key (K1)
101 disk serial number
102 process of validating disk serial number
103 bus input buffer
104 decrypted program(s) and/or data in SRAM
105 bus output buffer
106 process of authenticating programs/data
107 process of RSA decryption
108 process of calculating hash values
109 key selection number
110 table of keys
111 process of block decrypting (K1)
112 hash value
113 key selection number
114 game product number
115 internal data bus
116 internal address bus
117 video signal generator
119 LCD driver
120 game vendor's server
121 the Internet data transmission network

122 game software database
123 process of reading database
124 encrypted key (K1) and other data
128 processor cache memory
129 process of block encryption (K2)
130 electric battery or cell
131 secret key (K3)
132 enhancement software in RAM B
133 process of block encryption (K1)
134 processor core and ROM in crypto chip 52
136 verify game id or checksum
137 sending modem (modulator)
138 receiving modem (demodulator)
139 crypto chip identifier
140 address table in cache
141 enhancement number
142 process of block decryption (K4)
143 file of disk serial numbers or chip identifiers
144 RSA encrypted hash value
145 hash value
147 process of block encryption (K4)
148 lead-in control data (encrypted key)
149 disk molding machine
150 process of burning BCA into disk
166 RSA private key
167 RSA encryption process
230 insertion socket for cartridge
239 SRAM shared
247 multiple contact connector
279 multiple contact connector
299 cartridge circuit board
300 direct memory access (DMA)
301 image coprocessor
302 video RAM
303 crypto processor
304 session key (K4)

306 process of block encryption (K3)
307 process of block decryption (K3)
310 thermal noise source
311 generate session key
313 boot ROM in crypto processor 303
314 response timer
323 encrypted chip identifier
324 retailer computer